Computer Sciences Department

Regression, Regularization, and Redundancy: Humans' Response to Redundant Inputs in a Linear System

Rachael McCormick

Technical Report #1704

October 2011

UNIVERSITY OF WISCONSIN MADISON

Regression, Regularization, and Redundancy:

Humans' Response to Redundant Inputs in a Linear System

Rachael A. McCormick

University of Wisconsin – Madison

Abstract

In this study, I explored the affect redundant or highly intercorrelated input features had on human participants' ability to learn a linear regression-type task. Earlier studies suggest that, paradoxically, people perform *worse* with redundant input, something which could possibly be explaining by using regularization to sacrifice training set accuracy for model generalizability. I introduce a novel paradigm for having humans perform linear regression, for calculating what β weights they learned, and for establishing whether they favored the non-sparse L_2 or the sparse L_1 regularizer. I found that people form into two distinct groups, on favoring a sparse strategy and the other favoring a non-sparse strategy, but was not able to manipulate which strategy participants adopted. Discussion included implications for psychological and machine learning research.

Special Thanks
to

Letters & Science Honors Program

for their generous funding through the
Honors Senior Thesis Summer Research Grant

Air Force Office of Scientific Research

for their funding and support through grant
AFOSR FA9550-09-1-0313

My Advisors,

Timothy T. Rogers
Deptarment of Psychology

Xiaojin (Jerry) Zhu
Departments of Computer Science and Psychology

for their insight and guidance,

And My Boyfriend,

Nate Peterson
for his support, patience, and confidence.

Contents

Introduction

Judgment tasks make up an enormous portion of human cognition: they cover many task areas, such as recognizing that the animal at your feet is your neighbor's dog, guessing the rough age of the person you are talking to, or learning what foods are safe to eat. The common form of judgment tasks is that they involve using some kind of input (often perceptual, but not always), and using some sort of rule which relates those inputs to produce a judgment. For example, you might learn that your neighbor's dog has a certain spotted pattern on his coat, is a certain size, and has a certain pitch to his bark. Even if you only have partial information, such as seeing a small animal of about the right size in the dark and hear a bark of that pitch, you are able to judge with reasonable certainty that the creature is indeed your neighbor's dog. This is because you have learned to use various input cues to make judgments about what animal it is that you are interacting with.

There are *many* ways to model how humans perform judgment tasks. One of the simplest is *linear regression*. "Regression" is simply a subset of judgment tasks that involve continuous-valued decisions (such as guessing just how heavy something will be just by looking at it and knowing what it's made of), as opposed to categorical decisions (such as whether an animal is one's neighbor's dog or a raccoon). *Linear* regression is so-named because it makes a simplifying assumption about the relationship between the inputs and the judgment to be made: that the relationship is linear. Specifically, linear regression is the act of finding the best *linear model* to describe the judgment task at hand, and a linear model is simply a weighted sum of the input features:

$$judgment = weight_1 * input_1 + weight_2 * input_2 + \ ... \ + weight_N * input_N \quad (1)$$

The "weights," also called β weights or β values, represent how large of an effect that particular input feature has on the final decision. (Especially if the inputs are standardized, so their values are all in the same range.) Linear models *cannot* capture non-linear relationships, such as "when $input_1$ is above 5, $input_2$ doesn't matter (its weight is 0)." One notable side-effect of linear models is that all input features are assumed to be independent, which is often not perfectly true.

Despite its great simplicity, linear regression plays a major role in machine learning, as it is a very basic and easily-adaptable mathematical formulation of learning. For example, in addition to linear regression being used on its own to train computers to make judgments, it is also a key component of the widely used and successful neural network and support vector machine algorithms. The linear model also has a large role in cognitive psychological research. Besides using linear regression as a form of statistical analyses, linear models have been used in attempts to mathematically describe humans' decision-making behavior, such as Egon Brunswik's seminal "lens model," which uses precisely Equation 1 to model the judgments of human subjects on decision-making tasks (Karelaia & Hogarth, 2008). Many factors have been identified which contribute to humans' ability to learn the optimal linear model for a classification task (i.e., the model which makes the fewest erroneous classifications), such as number of features and participant expertise with the decision domain.

One factor whose effect on learning is still unclear is feature intercorrelation and redundancy. (Or, when the input features are *not* perfectly independent.) Features are considered *strictly redundant* if they convey identical information, albeit in a different form. For example, imagine that for a particular variety of apple, larger fruits are *always* more red and vice versa. If you know that the more red the fruit is, the tastier it will be, than you also know that the larger the fruit is, the tastier it will be. Therefore, when trying to select the best apple to eat, you could

look for the largest one without looking at their colors, or vice versa—size and color provide redundant information. Even if the rule about larger apples being more red isn't always perfectly true—you could, given a lot of apples, find one that is more red than an apple larger than it—using just color or just size to determine tastiness could still be a sound rule of thumb. If the rule about size and color isn't perfectly true, we can still say these two features of apples are "intercorrelated"—larger apples *tend* to be more red, and vice versa. Redundant and intercorrelated features are common in naturalistic problems, and humans deal with them regularly when learning new tasks. It would be valuable, then, to more clearly understand how redundancy affects people's learning.

Brunswik originally proposed that the mutual substitutability of redundant features would improve the reliability of judgments and improve response time (Brunswik, 1955). Since then, however, numerous studies have found the opposite effect; a recent meta-analysis spanning over 50 years and 249 classification-task experiments testing the lens model found that overall, increased feature redundancy decreased judgment accuracy, reliability, and subjects' responsiveness to instructional feedback on the task (Karelaia & Hogarth, 2008). These results directly oppose machine learning theory, which states that a linear model could never perform *worse* with strictly redundant features, because one can always at worst just treat the redundant features as independent and process them as per usual. Identifying their redundancy is only a "shortcut" which allows the learner to ignore a feature and save on processing time somewhat.

There is a possible explanation, however, for why people might perform *worse* with redundant features: regression can be "regularized," a method which effectively reduces the learner's accuracy on the examples they are given to learn from, but (hopefully) improves their performance on examples they did not get to learn with. This is helpful if the features are only

redundant in the examples the learner has to learn from initially, but this is *not* true of the general population of examples. For example, a devious teacher could carefully select apples for which the rule about larger apples being more red is always true. From this biased sample, the learner will discover that it doesn't matter whether they use color or size to judge which apple will taste best. However, it could secretly be the case that only the color predicts the tastiness, and the teacher just did not show the learner any small red (tasty) or large green (sour) apples. Regularization is helpful because it assumes that the sample used for learning will not be a perfect representative of the population overall. (For a precise explanation of how regularization accomplishes this, see "Regularization" in "Regression, Regularizers, and Redundancy.")

It is unlikely that people normally encounter these sorts of purposefully-biased samples, but it is true that the fewer examples one learns from, the more likely this sample is to be biased, as per the law of large numbers. Humans are exceptionally able to learn from small samples sizes, suggesting that they may employ various mechanisms, such as regularization, to boost their performance assuming they have a somewhat biased sample to learn from. They may be particularly inclined to regularize their learning when they detect that some of the features are redundant, naturally "suspicious" of any apparent redundancy. (Deciding too soon to completely ignore one of the redundant features may mean you do not notice later on when you see examples that violated the redundancy, like a small tasty red apple.) However, regularization *does* sacrifice some performance on the learning examples, and in the case of many psychology experiments, the learning sample is carefully crafted to *not* be biased, meaning regularization would just hurt performance in these experimental cases.

Therefore, in order to better understand the effects of feature redundancy on human decision-making, I experimentally tested what regularization method (if any) people appear to

use when given redundant features to learn from. I hope that this will help explain the curious finding that redundant features impair people's learning. I chose to use a continuous-valued judgment task (e.g. "how tasty will this apple be?") as opposed to a classification task (e.g. "is this a tasty apple or a sour one?") because continuous-valued decisions are a more general form of the problem. Continuous judgments can be made into classification judgments with simple thresholding, such as deciding that above a certain cutoff an apple is deemed "tasty" and below it it is "sour." Continuous-valued judgments are (mathematically) simpler to describe and reverse-engineer, because they do not involve this thresholding step.

Regression, Regularizers, and Redundancy

Linear Regression. Linear regression is the process of developing a linear model to describe relationships in the data. A linear model is essentially the "rule" the learner develops in continuous-valued judgment tasks, and simply predicts the output measure for a given example through a weighted sum of the continuous-valued input features; these weights are called the beta weights (β). In the apple example, size (perhaps in ounces) and color (as a continuous hue measure along the rainbow) are the inputs to the regression analysis (x_1 and x_2), and tastiness (on some fictional scale) is the output measure (y). The resulting linear model, which calculates the *estimated* output $\hat{y}_i$ for a given example i, can be described as follows (where x_{fi} is the value of the f^{th} input feature for example i, and ε_i is the noise term for i (giving room for imperfect or 'noisy' inputs)):

$$\hat{y}_i = \sum_{f=1}^{\#\,features} \left(\beta_f\, x_{fi}\right) + \varepsilon_i \qquad (2)$$

Generally, when developing a linear model, the goal of the regression is to choose β values that minimize the difference between the estimated output $\hat{y}_i$ and the true output y_i. Regression takes in the values for all the input features for all the examples in its *training set* a well as the output values for all those examples, and attempts to produce the best set of β weights possible. The difference between $\hat{y}_i$ and y_i is usually measured as the sum squared error (SSE) over all the training set examples. Therefore, the work of the regression analysis is to solve the following expression (which adjusts β, the weight vector, to minimize the SSE):

$$\min_{\beta} \left[\sum_{i=1}^{\#\, examples} |y_i - \hat{y}_i|^2 \right] = \min_{\beta} [\, SSE \,] \qquad (3)$$

The Problem of Overfitting. One problem which all machine learning algorithms must deal with is *overfitting*, which is when the learning algorithm maximizes its accuracy on its given training set at the cost of *generalizability* to other data. One example is when the learner memorizes its training data "rote"—for example, an image classifier might just learn that images 1, 3 and 4 are of cats and 2, 5 and 6 are of flowers, rather than learning more in general what a cat looks like compared to a flower. Then, when other pictures not in its training set are shown to it, it responds at random, having never actually learned any sort of generalizable rule.

One common approach to counter overfitting is to subdivide the provided data, perhaps setting aside one tenth of the examples as a "tuning set" and using the rest normally as a training set. The algorithm learns using only the training set data, but then is tested periodically using the tuning data. Learning is halted when performance on the tuning set begins to *fall* relative to earlier testing periods: this indicates that the algorithm is beginning to overfit its training data, losing generalizability to data it did not train with (i.e. the tuning set).

A further step is *cross-validation*. For instance, one could repeat the above tuning process ten times, each time with a different tenth of the examples set aside as the tuning set. The learner with the best final accuracy on its tuning set is selected as the final model. (This would be "10-fold" cross-validation, which is considered the standard number of splits, or folds, in machine learning.) Cross-validation helps ensure that there are no random artifacts in the selection of the tuning set (e.g., all the very "easy" examples ended up in the tuning set) affect the final model.

Regularization. Vanilla linear regression (as described above) seeks only to find the set of βs which minimize the linear model's error for all the training examples it is given. This of course is very likely to cause overfitting. One common way of preventing linear regression from overfitting is to introduce a regularizer term to the minimized expression:

$$\min_{\beta} \left[SSE + \lambda * [regularizer]\right] \quad (4)$$

Depending on λ, the balance term between how important it is to minimize the SSE versus the regularizer term, this causes the regression analysis to sacrifice having a minimal error to some extent in exchange for greater generalizability beyond its training data. How exactly "greater generalizability" is achieved depends on which regularizer is used.

One notable family of regularizers is the L_p regularizers, so-named because they are based on the Lebesgue *p*-norms of the β weights. The original, the L_2 regularizer (also called ridge regression or the Euclidean norm), gives preference to choosing the smallest βs possible, by minimizing (the square root of) their squares:

$$L_2 = \sqrt{\sum_{f=1}^{\#\,feat.s} {\beta_f}^2} \quad (5)$$

Conversely, the L_0 regularizer (sometimes called the "minimum weight solution" or simply the zero-norm) is simply the count of how many *nonzero* βs there are in the model; therefore, minimizing it encourages using as few features as possible and setting all other redundant features' βs to 0. For this reason, L_0 is called a *sparse* regularizer. Unfortunately, calculating the L_0 regularizer term is NP-hard, or prohibitively time-consuming to compute (for a full proof, see Amaldi & Kann, 1998). While NP-hard problems should not necessarily be discounted, because humans can make decisions so quickly, this makes the L_0 regularizer *not* a likely candidate for directly modeling humans' behavior. The L_1 regularizer (also called LASSO or Manhattan distance norm) is a middle ground between L_0 and L_2, and is the sum of the absolute values of β:

$$L_1 = \sum_{f=1}^{\#\,feat.s} |\beta_f| \tag{6}$$

At first, this seems very similar to minimizing the squares of β (as in L_2); however, it actually behaves more like to L_0, acting as a sparse regularizer. While squaring the βs makes larger βs exponentially more costly, simply taking their absolute values means all βs are penalized linearly, causing the L_1 regularizer to favor fewer nonzero βs.

Furthermore, more than one regularizer term can be used in the regression analysis, each with their own λ. For example, one could use both the L_1 and L_2 regularizers:

$$\min_{\beta} \left[SSE + \lambda_1 L_1 + \lambda_2 L_2\right] \tag{7}$$

This specific hybrid model is called an *elastic net*, first described by Zou and Hastie (2005). Elastic nets have many interesting properties and strengths, including handling redundant features more effectively than either of the L_1 or L_2 do alone. (For further discussion of this, see "How Regularizers Handle Redundancy.")

The final question is how to select the appropriate λ(s). Clearly, $\lambda = 0$ will not use the regularizer at all and behave like vanilla (unregularized) regression. Conversely, an extremely high λ will dominate the SSE term, meaning the model will have no concern for finding βs which improve accuracy, only those which satisfying the regularizer. For example, a model with an overly-large λ for L_1 will only concern itself with having the fewest nonzero βs possible—it will quickly conclude setting all the weights to zero is the best solution!

It can be very difficult to predict what will be a good λ, so one solution is to *tune* λ by running the regression analysis repeatedly, slightly increasing λ each time. In order to tell which λ produced the best linear model, a portion of the examples can be set aside as a tuning set (as described in "The Problem of Overfitting"). After performing regression analysis with the training set of examples (which does *not* included the tuning examples), the linear model is used to find the predicted outputs ($\hat{y}$) for the tuning examples. The model with the best prediction accuracy is considered the best fit, and to have the most "correct" βs. Furthermore, cross validation should also be applied. Compared to a single run of vanilla linear regression, tuned and cross-validated regularized regression will take longer to compute, but will produce more accurate and meaningful βs, as they will also be relatively accurate for novel data, suggesting the better describe the true underlying relationships. (For those interested in applying this technique, the glmnet package for R (Friedman, Hastie, & Tibshirani, 2001) is a very fast implementation of tuned and cross-validated elastic net regression.)

Redundancy. Two features are considered redundant if they convey the same information about the output measure, even if their values are different. (Though identical features are by nature redundant.) For example, the diameter of a circle is always twice the radius (d = 2r), and

the area is $\pi r^2 = \frac{1}{4}\pi d^2$. If a regression task has both diameter and area as input features, while their values will rarely be identical, they are redundant: one can always find the area by knowing the diameter, and vice versa, meaning only one of them is actually needed. In this example, the features are *strictly redundant*, meaning they have a perfect correlation of 1.0. Strict redundancy is a specific subtype of *feature intercorrelation*, where features are significantly correlated, though not always perfectly. If the measurements of area and diameter were noisy, they would no longer be strictly redundant, though highly intercorrelated.

Strictly redundant, and even highly intercorrelated features pose trouble for regression analysis. If two features x_m and x_n are strictly redundant, only one is needed and no additional information is conveyed by the other. Therefore, the model can give weight to only x_m and set x_n's β to zero (effectively ignoring it), vice versa, or any intermediate combination of the two (e.g., weighting them equally). Notice that this gives rise to infinitely many possible weighting combinations for just two redundant features! More generally, the βs for two redundant features can be distributed any number of ways, so long as their *collective* weight, or their importance relative to other independent features in the model, is fixed. (Here, let β_{mn} be the collective weight given to features x_m and x_n, and p be a free variable from 0.0 to 1.0):

$$\beta_{mn} = p * \beta_m + (1 - p) * \beta_n \quad (8)$$

How Regularizers Handle Redundancy. Regularizers help offer a solution to which of the infinite solutions for β of redundant features the model should settle on. The L_2 regularizer encourages weighting all redundant features equally. Consider the most basic form of redundancy, two identical features: $L_2 = \sqrt{(0.5)^2 + (0.5)^2} \approx 0.71$ is less than any other potential combination. (The other extreme, completely ignoring one of the features, results in

$\sqrt{(1)^2 + (0)^2} \approx 1.0$. Note, however, how the L_2 term is not very different even at its two extremes!) Conversely, sparse regularizers such as L_1 will produce the opposite effect: one of the redundant features will be chosen to bear all their collective weight and the other(s) will be given $\beta = 0$. While there are not infinitely many sparse solutions for weighting redundant features, there are still as many as there are features, depending on which feature is chosen to have a nonzero weight. Therefore, L_1 is technically undefined for strictly redundant features, though any of the redundant features can be chosen arbitrarily as the weight-bearer to produce a solution.

Elastic net regression (equation 6), which uses both L_1 and L_2 regularizers, has interesting behavior given highly intercorrelated features, making it more effective in these situations than using L_1 or L_2 alone (Zou & Hastie, 2005). Specifically, if λ_1 is substantially greater than λ_2, there is a "grouping effect" where highly intercorrelated features are effectively averaged together and their average is then entered into the model as one feature, but overall a sparse model is still produced. This is because, when λ_1 is high, the model generally prefers sparse solutions, but when it does encounter the situation where the features are redundant and the optimal weighting solution is undefined, it reverts to L_2 regularization to break the tie, as L_1 regularization has no preference between the redundant features.

Regularization by Humans

Given that there are multiple ways of handling strictly redundant features in a linear system, and that the presence of redundant features clearly affects human subjects' performance on decision tasks, what "regularizer" (or combination thereof) are people using? It is quite likely that humans use *some* form of regularization for continuous-valued judgment tasks (i.e. regression tasks). Intercorrelated features and noisy inputs are common to human experience,

and are things regularizers help with. Furthermore, humans do *very* well at avoiding overfitting: we are naturals at identifying general patterns and rules, and often do poorly when trying to memorize various facts "rote." While they likely do not perfectly follow the mathematical formulation of any of the L_p regularizers, one can compare human behavior to the general "strategies" of the various regularizers: do humans prefer to consider all redundant features more or less equally, or do they prefer a more sparse solution?

There are advantages and disadvantages to both approaches: sparse solutions are clearly faster and simpler to use, as fewer features need to be considered. However, they offer less insulation against noisy inputs: as soon as noise is introduced to the system (and noise is present in *all* sensory experiences for humans), even redundant features are no longer perfectly correlated. Instead, computing the average of these features actually helps get at the underlying *true* (noiseless) value of the features, much like taking repeated measures. Therefore, in noisy situations, less sparse regularizers offer a distinct advantage over sparse ones, which will be affected greatly by the noise since they attend to only one of the redundant features.

To date, experimental evidence for which regularizer humans most closely match is very mixed and inconclusive. Some studies have found that people try considering all features equally (Schmitt & Dudycha, 1975), while others have found that people employ a "take the best" heuristic and prefer to examine as few features as possible (Pishkin & Williams, 1984; Hutchinson & Gigerenzer, 2005). Furthermore, I was not able to find any studies that have explored human "regularization" in conjunction with (strictly) redundant features.

Regularizers offer formally explicit hypotheses about how people might cope with redundant information in a continuous, linear regression-like setting; for example, the L_1 and L_2 regularizers contrast sparse and non-sparse (equal-weighting) solutions, and hybrid systems like

the elastic net provide even more detailed predictions for responses to redundancy. This makes them a good framework for investigating and explaining how people use redundant data.

Furthermore, while machine learning makes extensive use of linear regression and related models, rarely are models' designs informed by human behavior. For example, the best λ multiplier(s) for the regularizer term(s) are arrived at through tuning and cross-validation, not by considering what a human would prefer. However, it is highly unlikely that humans run the same regression analysis thousands of times in their heads, tuning their λs! Instead, it is possible that humans somehow know what strategies (i.e. degree of sparsity) are preferable in different situations, and are able to adjust their analysis on the fly. Knowing how to design such dynamic linear analyzers could be of great interest to the machine learning community, especially the areas of extended and continuous learning.

The Present Study

In this paper, I report the results of a series of three experiments designed to investigate human behavior in a simple linear regression task with redundant features. The first experiment introduces the new paradigm I developed, as I could find none to replicate which could address our questions. It involves very simple conditions to test whether people can learn the task in our paradigm, and verifies that our novel methods for measuring peoples' β are accurate. The second experiment uses the same paradigm and analysis methods, but involves a much more difficult input set that prevents people from using obvious superficial solutions. This experiment also introduces a method of determining which regularizers best matches participants' behavior. Finally, the third experiment replicates the second except with noisy inputs, allowing us to investigate whether people adjust their regularization strategies depending on the situation at

hand. I hypothesize that with noisier inputs, people will tend towards less sparse β solutions than in the noiseless version, as non-sparse regularizers handle noise better than sparse regularizers (see "Regularization by Humans").

Experiment 1

A Paradigm for Studying Human Regression

I could not find any previously-used paradigms for testing human regression capability, especially using fully continuous-valued inputs and outputs and allowing for strict redundancy between features. Therefore, I developed my own, built around the simple back story that the participants are paleontologists-in-training, learning to use a chemical marker to determine the age of fossils. The "catch," participants are told, is that the level of the chemical is different in three different parts of the fossils' bodies, and some parts are more informative in determining the age of the fossil than others.

The paradigm involves a learning (or "training") phase followed by a testing phase, composed of 62 and 23 separate trials, respectively. Each trial represents a different fossil. Participants have a limited amount of time to look over the three chemical levels (represented as partially-filled in meters) and respond with their estimated age for that fossil. (For a more precise explanation of the program, see "Methods: Procedure.") In the training phase, after they submit their answer, they see their answer alongside the true age as well as the three chemical meters, allowing them to adjust their mental model (and hopefully learn) before moving on to the next fossil. In the testing phase, however, they are not shown the correct answer, but instead go on to the next trial immediately.

Participants are encouraged to improve their performance by tracking their error (absolute difference between their estimated age and the true age). If they kept their error below a certain threshold for 20 trials, they were said to have reached criterion, and went on to the testing phase. If a participant never reached criterion, they would begin the testing phase after completing all 62 training trials.

In terms of linear regression, the chemical levels are the input features (x_1, x_2 and x_3), and the age of the fossil is the output measure. Both the input and outputs are continuous-valued, and can be negative or positive. The age of a given fossil i is calculated from the three chemical levels—how "informative" it is for determining the age is essentially its β:

$$age_i = y_i = \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} \quad (9)$$

Using this paradigm, any set of β weights and input x values can be chosen, and the matching y values derived using this equation. To produce redundant inputs (x_1 and x_2) and one independent input (x_3), x_1 and x_3 are chosen randomly (uniform in the range (-1, 1)) and x_2 is derived from x_1 using a linear relationship:

$$\begin{gathered} x_1, x_3 = uniform(-1,1) \\ x_2 = a * x_1 + b \end{gathered} \quad (10)$$

Two input features in this way will clearly not have identical values, but will be strictly redundant: given knowledge of x_1, one can always calculate what x_2 must be and vice versa. The linear relationship is a relatively simple one, but the scaling (a) and offset (b) serve two purposes: First, to obscure the relationship between x_1 and x_2, so participants do not readily notice that they are in fact strictly redundant. In Experiment 1, b was always -0.4, but a was either +0.7 or -0.7 on a between-participants basis (see Table 1 for a summary of different experimental conditions). These two levels were chosen to see if *how obscure* the redundancy was would have any impact on people's behavior, with the "negative" condition (a = -0.7) considered the "more obscure" one, as the relationship between x_1 and x_2 is less clear when they are always of opposite sign (due to the negative multiplier)

Second, the offset term constrains the normally-infinite possible solutions for the βs of two strictly redundant features (see equation 7) to just one unique solution. Imagine plotting the

Table 1: Experimental Conditions

Experiment	β	a	b	Noise
1: positive	(1, 1, 1)	+0.7	-0.4	no
negative	(1, 1, 1)	-0.7	-0.4	no
2:	(0.55, 1, 1.6)	-0.7	-0.4	no
3:	(0.55, 1, 1.6)	-0.7	-0.4	yes

β weights for two features (say x_i and x_j) with β_i on one axis and β_j on the other: generally, the correct β would be a single point on the graph—this is a *point solution*, where there is a exactly one unique solution for β. However, if x_i and x_j are strictly redundant and, for example, one is a simple rescaling of the other (e.g. $x_i = a * x_j$), the solution for the correct β can be drawn as a straight line: any of the infinite points along that line are functionally identical. This is called a *line solution* for β.

Unfortunately, it is harder to tell if participants' own βs conform to the correct line solution than if it conforms to a point solution. Consider a scatter plot of participants' βs, with the points representing the participants distributed in a loose cloud. With a single point as the correct beta, it is easy enough to determine if participants' βs crowd around this point or are distributed more or less randomly. But if the correct answer *line* bisects the cloud, it is much harder to determine if the cloud clusters meaningfully towards the line. Fortunately, the offset term forces there to be one unique point solution to the system. In other words, the β I select is *the* correct answer, and I can easily compare participants' βs to this.

Experiment 1 has two goals: to verify if people can learn our task, and then to determine if our novel method for approximating participants' βs is valid. For this reason, I chose a very simple scenario, with $\beta_{1,2,3}$ = (1, 1, 1) (see Table 1 for experimental conditions.) This means all three input features are equally important (since they have the same βs), and furthermore, the

correct answer for the age is $y_i = 1 * x_1 + 1 * x_2 + 1 * x_3 = x_1 + x_2 + x_3$, or simply the sum of the three chemical levels. This is the "easiest" β that still uses all three features, as it requires no feature selection (determining what features are irrelevant, or have a $\beta = 0$) nor differential weighting between features.

Methods

Participants. The participants were 27 undergraduate students (14 females, 13 males) enrolled in an introductory psychology course at the University of Wisconsin – Madison. Ages ranged from 18 to 21 (M = 19.04, SD = 0.98 years). Participation was voluntary, and participants received extra credit for their course for their time. Participants were treated in accordance with the APA's "Ethical Principals of Psychologists and Code of Conduct" (American Psychological Association, 2002).

Materials. The study was conducted using a program written in Java by the author for this purpose. The program was run on Dell computers with Windows XP, Intel Celeron 2.4 GHz processors, 256 MB of RAM, and 17 inch CRT monitors at their native resolution of 1024x768. There are six such identical computers in a small computer room, where participants were run in batches of up to six (M = 5.63, SD = 0.49 simultaneous participants). After the program finished, participants were asked to fill out a brief questionnaire which asked them to describe the general "rule" they developed for determining their answers, to rate each meter's "importance" in determining their answer on a scale of 0 to 100, and finally, the specific rules they had for each of the three meters (for a copy, see Appendix B).

Procedure. Participants were assigned randomly to either the "positive" or "negative" condition. Before beginning the experiment, all participants read and signed consent forms. Then

the experimenter read them the experimental protocol (Appendix A) which introduces the story, in which they are paleontologists-in-training, and gives an overview of the procedure and controls of the computer program. After this, they are again shown directions on the screen which they may read through at their own rate before beginning the training phase.

In the training phase, all participants had the exact same 62 trials, and progressed through them in the same order. For each trial, participants are first shown a screen (Figure 1) with a crocodile fossil in the background and three small meters with lines connecting them to the fossil's head, torso and limb. These meters represent the levels of the chemical marker found in these parts of the body, and are filled with red coloring to different levels for each example. Negative values are represented by the red coloring extending below the midline, while it extends above the midline for positive values. There are no ends drawn on the meters, but rather they fade away, to suggest that the values are unbounded, but this is the range of interest. The meters do have ruler-style tick marks along the side, to aide with height measurement. The red meters range from -1 to +1.

Superimposed over the fossil in the center is a fourth, much larger meter—the level of this meter is set by the participant using the keyboard, and represents their estimated age for the fossil. It is designed just like the smaller input meters, though its fill color is blue, and according to the tick marks, it ranges from -6 to +6. Finally, the screen also shows a timer in the lower left, which begins at 10 seconds and ticks down to 0, and an "error meter" off to the far right. The height of the error meter represents the error they have accrued so far, and the number beneath it shows how many turns it has been since the meter reset. If they reach 20 turns without filling the meter, they finish the training phase.

Figure 1: Response Screen

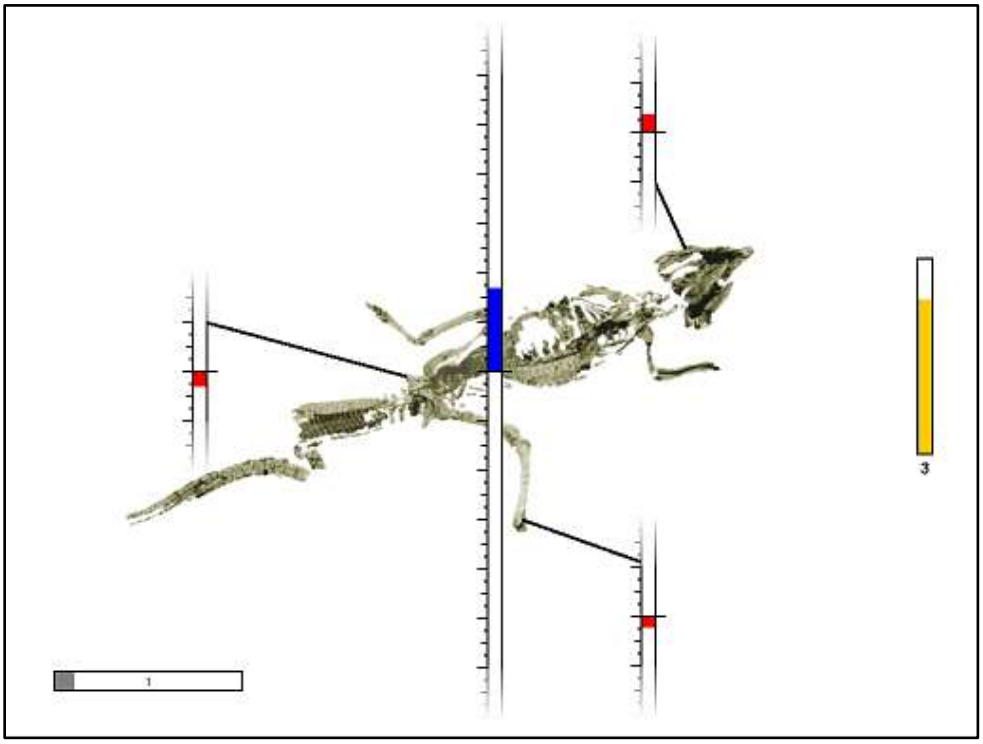
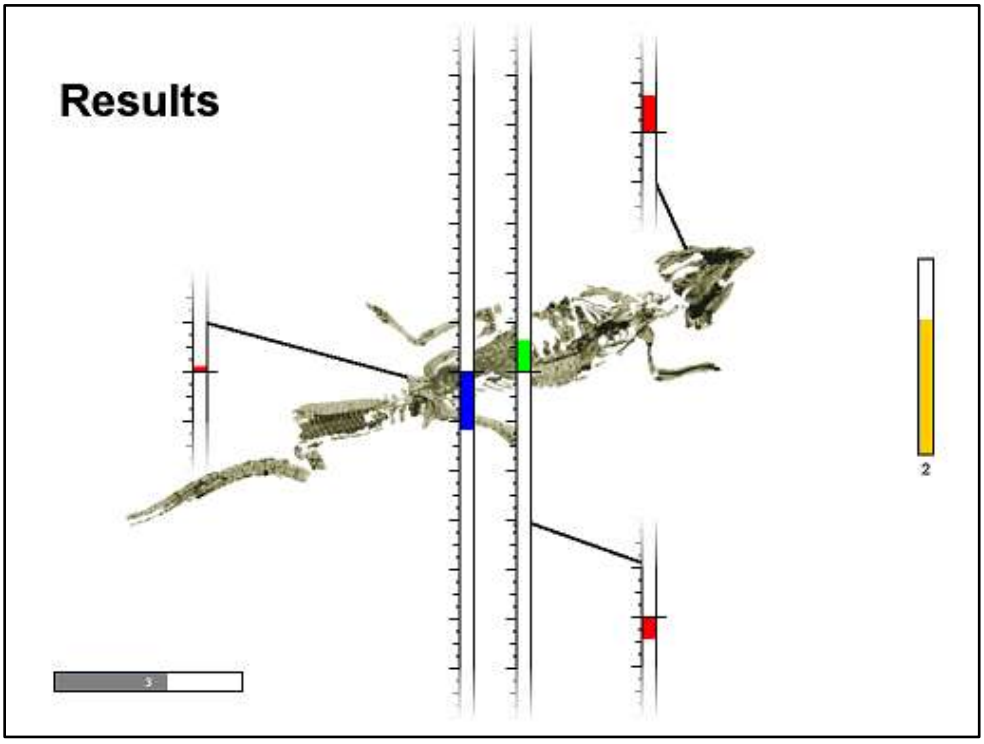


Figure 2: Results Screen



After 10 s, the screen changes, accepting whatever the participant had set the central blue meter to as their answer. The next screen (Figure 2) is very similar to the first, except it has two large meters in the second—the left shows what answer the participant submitted, and the right shows the correct age of the fossil using a green fill color. Furthermore, the magnitude (or absolute value) of the difference between their blue answer meter and the green correct answer meter is added to the orange-colored error thermometer on the right, and the number beneath the meter incremented by one. After 5 s (again displayed on the timer), if they have managed to go 20 turns without filling the error meter (thus reaching the learning criterion) or if 62 training trials have passed, the training phase ends and the testing phase begins. Otherwise, an inter-trial screen reading "Preparing next fossil…" is shown for 2 s and then the process beings again.

Error criterion for all conditions was defined as 4.0: if a participant's accumulated absolute error stayed below 4.0 for 20 trials, they "reached criterion." If they did accumulate 4.0 total error, filling the error meter, their accumulated total was reset to 0, but so was their count towards the 20 trials. (Visually, the meter would empty, and the number reset to 0.)

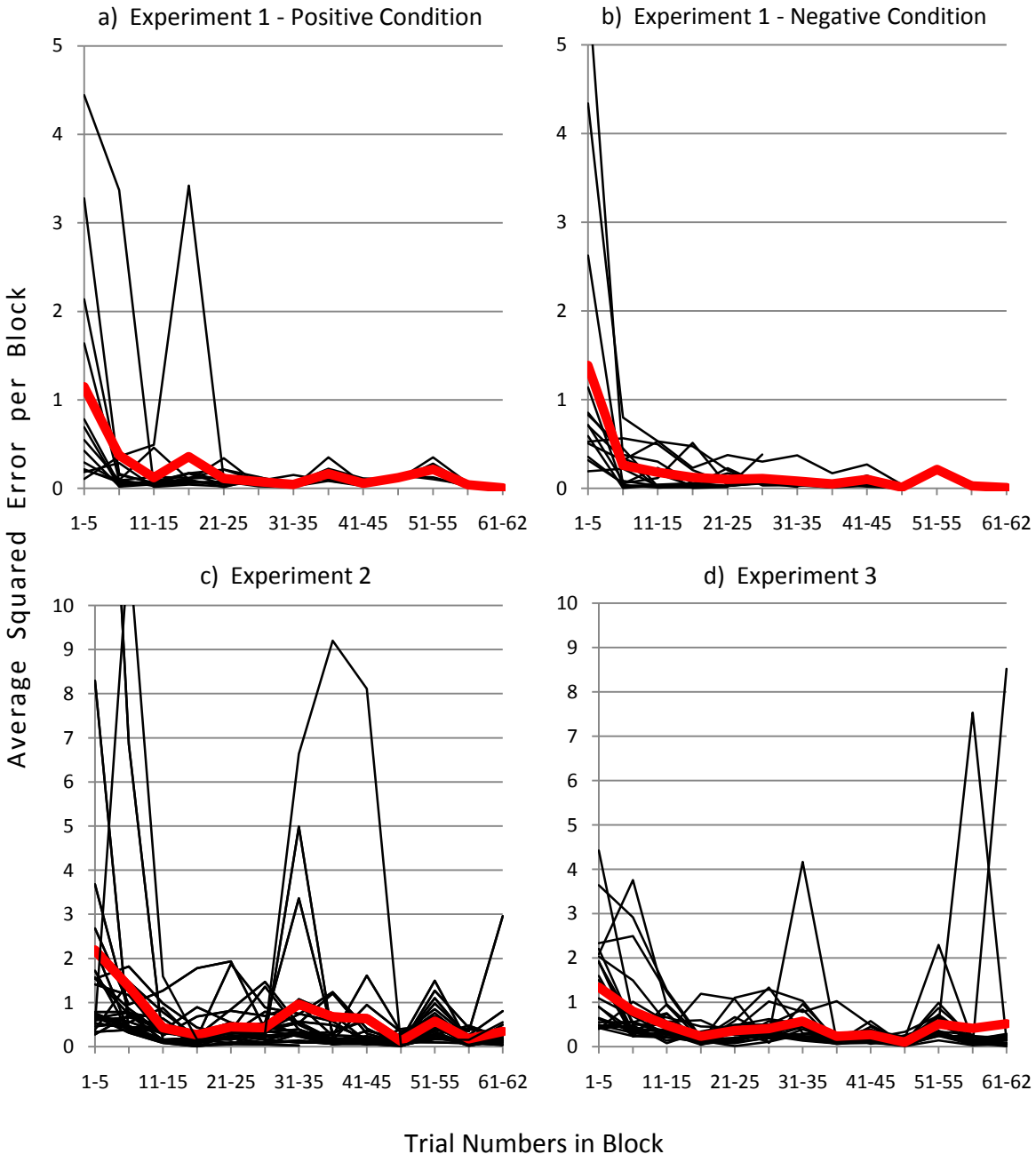
For participants, the testing phase is nearly identical to the training phase, except the correct answer screen is *not* shown after the guess submission screen. (The testing phase is called the "final exam" of the paleontology training, justifying why they do not receive immediate feedback on their performance.) Also, the error meter is not shown, as all participants will be doing all 23 trials regardless of their performance. After completing the testing phase, participants are given the questionnaire and instructed to take their time answering it. After they finish, they are thanked and debriefed.

Results

No effect of participant age, sex, study time, number of participants run simultaneously, or which computer participants used was found on any of the following analyses. Non-learners were defined as those who did not achieve a sum squared error (SSE) of less than 12 in the testing phase: using this standard, all the participants in both conditions qualified as learners, and were included in the following analyses.

Performance. In order to view performance over time, in the training phase accuracy is reported as the average squared error for a block of five trials: $average_{i\,\epsilon\,block}\left(\left|y_i - \hat{y}_i\right|^2\right)$ (Figure 3). This measure is less sensitive than sum squared error (SSE) to the occasional extremely poor trials some participants had. Additionally, many participants in both conditions reached criterion and finished training before the 62nd trial; however, there was no difference in the number of training trials completed by the positive (M = 44.92, SD = 12.67) and negative conditions (M = 39.79, SD = 12.05) (t = 1.08, p > .1). In contrast, performance in the testing phase was measured using the SSE for all 23 test trials. Participants in the negative condition (M = 0.90, SD = 0.78) performed better (had lower SSE) in the testing phase than those in the positive condition (M = 2.35, SD = 1.61) (t = 2.92, p = .01).

Figure 3: Training Phase Performance



The average squared error in each block of five training trials, depicting learning over time. Each black line is a different participant, and the red line is the average of all the participants. If a participant stopped the training phase early because they reached criterion, then their line ends where they finished training. All the large spikes are due to a single trial with extremely high error, in which the participant's answer was at or near the minimum or maximum response value (-6 or 6 respectively). Whether these trials are errors is uncertain, but they did not impact test phase performance.

A Method for Estimating Human β. In our paradigm, I developed a novel was of measuring participants' β weights, using paired trial probes in the testing phase. Specifically, if two trials are engineered to have identical input values except for one feature, then the β for that feature can be deduced from the difference in the participant's responses to these paired trials. Consider the paired trials *j* and *k*, whose only difference that some Δ is added x_1 in *k*:

$$\begin{aligned} y_j &= \beta_1 * x_{1j} + \beta_2 * x_{2j} + \beta_3 * x_{3j} \\ y_k &= \beta_1 * (\Delta + x_{1j}) + \beta_2 * x_{2j} + \beta_3 * x_{3j} \end{aligned} \tag{11}$$

It can easily be seen how comparing y_j and y_k reveals the value of β_1:

$$\begin{aligned} y_k &= \beta_1 * \Delta + \beta_1 * x_{1j} + \beta_2 * x_{2j} + \beta_3 * x_{3j} \\ y_k &= \beta_1 * \Delta + y_j \\ \beta_1 &= (y_k - y_j)/\Delta \end{aligned} \tag{12}$$

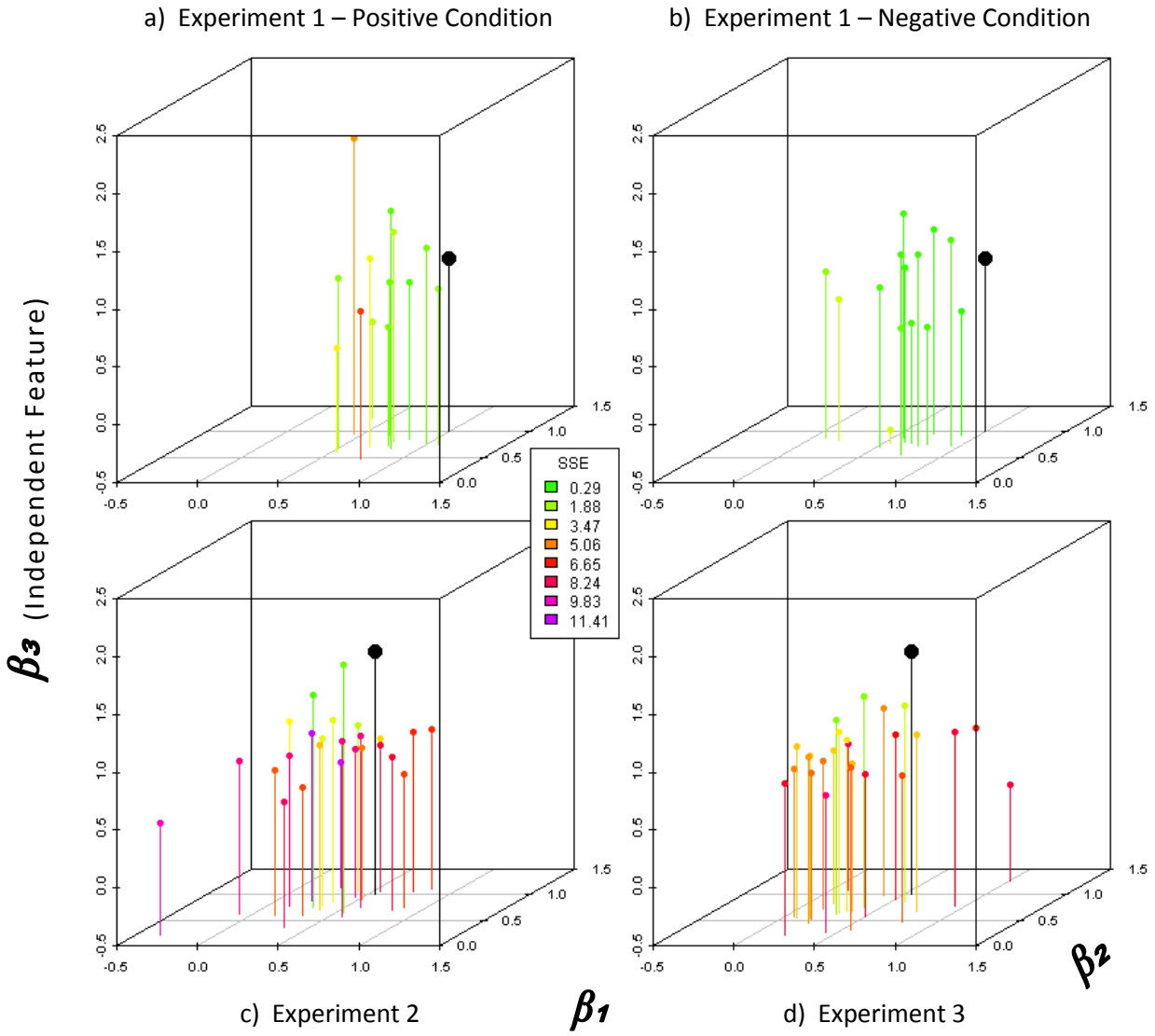
This paired trial system creates two key reasons for *not* providing feedback in the testing phase. (In addition to these reasons, I also want to "freeze" participants' models and discourage further learning midway through the testing phase.) As Equation 11 shows, a clever learner could easily use this very method to deduce the true value of β_1 if they were given the true output values (y_j and y_k). Furthermore, notice that these paired trials violate the strict redundancy relationship between x_1 and x_2 (Equation 9) used in the training phase. Because no feedback is given, it is our hope that participants will not notice that the old redundancy is not always true: if they were relying on this redundancy (i.e. using a sparse solution and ignoring one of x_1 or x_2), they would notice very quickly if they suddenly fail trials that vary the feature they were ignoring. (They would respond with identical $\hat{y}_j$ and $\hat{y}_k$, which is incorrect.) Without feedback, there is less chance that they will notice the change, especially because the redundancy relationship between x_1 and x_2 is already relatively obscure.

It is important to not present the second paired trial immediately after the first, or participants will employ their powerful change detection abilities to notice that only one value changed. However, the second trial should still follow relatively closely after the first, in case participants are still adjusting their models over the course of the testing phase. I arranged testing trials into blocks of six, with the first half of the paired trial probes for β_1, β_2, and β_3 (respectively) coming first, followed by the second halves of the pairs in the same order. While a minimum of one six-trial block would be required to probe all three βs, because human responses are by nature noisy, I had four blocks in the testing phase. All reported βs are simply the average of the four deduced values for that β.

Estimated Human β. Each participant's βs can be thought of as a point in a three-dimensional space of possible β values; thus, they are easily visualized in a 3D scatter plot (Figure 4). Because the correct β is a point solution, participants whose β points crowd nearest to the true β are those who best solved the linear system. A qualitative measure for how well the group as a whole managed to match that point solution is set of three one-sample *t* tests, one for each dimension (β_1, β_2, and β_3), using the correct β as their target. Using this measure, in the positive condition, participants' β_1 was lower than the true β_1 (M = 0.71, SD = 0.21, t = -5.09, p < .001), as were their β_2s (M = 0.79, SD = 0.19, t = -3.99, p = .002); however, their β_3s were correct—they did not vary significantly from the true β_3 (M = 0.98, SD = 0.48, t = -0.16, p > .1). The same pattern arose in the negative condition, with participants' β_1 (M = 0.61, SD = 0.24, t = -6.06, p < .001) and β_2 (M = 0.79, SD = 0.10, t = -7.79, p < .001) both being lower than their true values, while their β_3s are on target (M = 0.83, SD = 0.47, t = -1.37, p > .1).

The two conditions can be compared in a similar fashion, using an independent samples *t* test for each of β_1, β_2, and β_3. According to this comparison, there were no differences between

Figure 4: Estimated Human βs



Three-dimensional representation of participants' βs across all four experimental conditions. The large black point is the correct β. All other points represent individual participants, and are color-coded to indicate their SSE for the testing phase.

participants' βs in the two conditions (β_1: $t = 1.11$, $p > 0.1$; β_2: $t = 0.01$, $p > 0.1$; β_3: $t = 0.82$, $p > 0.1$).

Participants' estimated βs were not significantly correlated with testing phase SSE in the positive condition (all $r < .36$, all $p > .05$). In the negative condition, however, both estimated β_1 and β_3 were negatively correlated with test SSE ($r = -.65$, $p < .05$, and $r = -.59$, $p < .05$, respectively)—in other words, the lower participants' β_1 and β_3 were, the higher SSE was, so the *worse* they performed. These somewhat mixed findings may be an artifact of everyone doing so well, however.

Self-Report Data. After finishing the experiment, participants were asked to describe their rules for producing their answers, as well as rate the "importance" each of the three input meters has in determining their answers. While I performed no vigorous analysis of the self-report data, by simply looking through their responses, it is clear that in both conditions participants reported having an underlying β very much like (1,1,1). Their rules included such statements as "I added up the length of the bars (under the line = negative, above = positive) and filled the age bar according to scale," and "Adding all the bars together without any special importance for particular bars." Their importance ratings almost always featured identical ratings for all three meters.

Discussion

This experiment demonstrates that people are able to understand the game and learn the underlying βs. Though their test performance is not perfect, and their calculated βs are (in the case of β_1 and β_2) different from the true β, this could be a sign of regularization rather than poor learning. Their self-reported rules overwhelmingly describe simply adding up the values of the three input meters with no scaling, strongly suggesting a β of (1,1,1). The discrepancy between

the estimated βs and the self-report data could indicate a lack of awareness of the slight amount of regularization applied.

The self-report data also raise an important point, however: when the true β is (1,1,1), the correct answer is to just "stack" the heights of the three input red meters into the blue response meter. No scaling is required, because the scales between the meters are the same (with about an inch per full "unit" on the tick marks). This underlying β was designed to make the problem easy for people to solve, but it's quite possible that it is *too* easy: it does not tell us whether people will be able to learn more complex (and realistic) underlying βs. Furthermore, participants may prefer this "stacking solution" even when it is not quite correct, because it is much easier than trying to scale one meter by 0.96 and another by 1.02, for example. While regression analysis does not find a β of (1,1,1) special in any way, stacking solutions may be treated differently by humans, almost like a third kind of regularizer: one preferring "easy" over small or non-zero weights.

There is evidence of regularization in the conflict between participants' estimated βs and their self reports. However, in order to try to estimate which regularizers participants might have been using, I would need to account for this third regularizer. Unfortunately, with this experiment's data alone, defining "easiness" would be pure conjecture. More importantly, these points may only be easy because of the way the game and paradigm are set up, and are mere artifacts. Therefore, after this experiment, I sought to avoid true βs that had a stacking solution, in hopes of avoiding areas of the space of possible βs where there is a clear "easy" answer that will bias participants.

Another strong reason for avoiding stacking solutions is that they also make any effects of the redundancy between x_1 and x_2 difficult to discern, as stacking them is probably much easier than learning to use just one by scaling it sufficiently. There is evidence of some effect of

the redundancy, as the redundancy relationship itself is the only thing which varied between conditions, and differences between the conditions were found. (Notably, participants took longer to learn in the negative relationship condition, but then went on to outperform those in the positive condition during the testing phase.) Unfortunately, nothing further can be understood about the effect of redundancy from this experiment.

Experiments 2 sought to eliminate the influence of these superficial stacking solutions by using a carefully chosen true β for which there is no nearby "easy" alternative. Using this new β, questions about human regularization and response to sparsity are explored in greater depth.

Experiment 2

As discussed in the conclusion of Experiment 1, it is important to avoid an underlying β that is a "stacking solution" (where the inputs can be added without any scaling) for three reasons: first, to simply find out if people can learn more complex true βs. Second, to enable us to better discern the effects the redundant relationship between x_1 and x_2 has on participants' learning. Third, and very importantly, if there is even a similar underlying β that is particularly easy (such as simple stacking), human participants may show a preference for the "easier" βs, as a form of regularization. However, I don't know what exactly will constitute "easy" for humans, so it is best to stay well clear of all stacking solutions in hopes of minimizing the effect of this third regularizer. Without this regularizer having a noticeable effect on participants' solutions, I can calculate to what extent people are using sparse (L_1) and non-sparse (L_2) regularization.

In order to find an underlying β sufficiently far from a simple stacking solution, I plotted all seven stacking solutions in 3D β space (like the space shown in the 3D scatter pots from Experiment 1), and chose an underlying β that was distant from all of them. (The seven stacking solutions include, in addition to (1,1,1), those using only two of the inputs (e.g. (1,1,0)) and only one of the inputs (e.g. (1,0,0)).) Using this method, I selected a true β of (0.55, 1, 1.6) for Experiment 2. Besides changing the true β, however, nothing else was changed between Experiments 1 and 2, in order to continue testing the same paradigm as developed before.

Methods

Participants. The participants were 27 undergraduate students (12 females, 15 males) enrolled in an introductory psychology course at the University of Wisconsin – Madison. Ages ranged from 18 to 22 (M = 19.41, SD = 1.01 years). Participation was voluntary, and participants received extra credit for their course for their time. Participants were treated in accordance with

the APA's "Ethical Principals of Psychologists and Code of Conduct" (American Psychological Association, 2002).

Materials. The program, computers, and questionnaire were the same as those used in Experiment 1. Once again, participants were run in batches of up to six (M = 4.85, SD = 0.91 simultaneous participants).

Procedure. The procedure was identical to that in Experiment 1, from the protocol read at the beginning, through the program itself, to the questionnaire at the end. The only difference was the input values: $x_{1,2,3}$ were generated the same way, though this time $x_2 = -0.7 * x_1 - 0.4$ for all participants instead of having separate positive and negative conditions. (The negative a value was chosen because it better obscures the redundant relationship between x_1 and x_2, and participants performed better on it in Experiment 1.) The true β was also (0.55, 1, 1.6) in this experiment, meaning that y was calculated using this new linear equation.

Results

No effect of participant age, study time, number of participants run simultaneously, or which computer participants used on any of the following analyses was found. An effect of sex was found for performance on the testing set, with males (M = 5.04, SD = 3.21) performing better (having lower SSE) than females (M = 7.47, SD = 2.43) (F = 4.59, p =.04). However, sex had no observed effect on participants' β weights, nor their λs (see "Determining the Human Regularizers"), suggesting that sex may not have had an impact on how participants actually learned and the solutions they developed.

Non-learners were again defined as those who did not achieve an SSE < 12 in the testing phase: using this standard, 2 participants were classified as non-learners and excluded from the following analyses, leaving n = 25. Exclusions of these participants is justified in that they never

established a "rule" for forming their answers, so any βs I would deduce for them are extremely noisy and would not represent a coherent solution to the linear system.

Performance. As in Experiment 1, accuracy over time in the training phase is reported using average squared error (Figure 3). Unlike in Experiment 1, only one participant reached criterion and finished training early. Performance in the testing phase was measured using the SSE for all 23 test trials (M = 6.31, SD = 3.04). Notice that, on average, participants performed worse in this experiment than in the easier conditions used in Experiment 1.

Estimated Human β. In this experiment, all of the participants' βs were lower than the true β (Figure 4). β_1 was below the true 0.55 (M = 0.39, SD = 0.29, t = -2.72, p = .012), β_2 was below the true 1.00 (M = 0.79, SD = 0.23, t = -4.38, $p < .001$), and β_3 was below the true 1.60 (M = 0.87, SD = 0.24, t = -15.03, $p < .001$). In this experiment, only participants' estimated β_3 was correlated with testing phase SSE ($r = -.62$, $p < .01$)—again, the correlation was negative, so the lower participants' β_3s were, the higher SSE was, or the *worse* they performed.

Determining the Human Regularizers. Clearly, participants' βs are notably different from the true β. This does not necessarily mean the participants are doing poorly on the task, however. As I discussed, sometimes perfect performance on the training set is not the best goal, and it is wise to regularize the linear equation to prevent overfitting and improve generalizability of the rule produced. While the true β, (0.55, 1, 1.6), is actually perfectly correct in this experiment (because the y values were generated *using* this β), the learner does not know that, meaning some form of regularization is still a reasonable choice. If regularized, linear regression analysis will produce points different from the true β; the point's distance from the true β will grow with the magnitude of λ, as less and less preference is given to training set accuracy and more to the regularizer term. In the 3D scatter plots, sparse regularizers (e.g. L_1) will prefer solutions where at least one β weight is 0—their points will tend towards the axes of the graph.

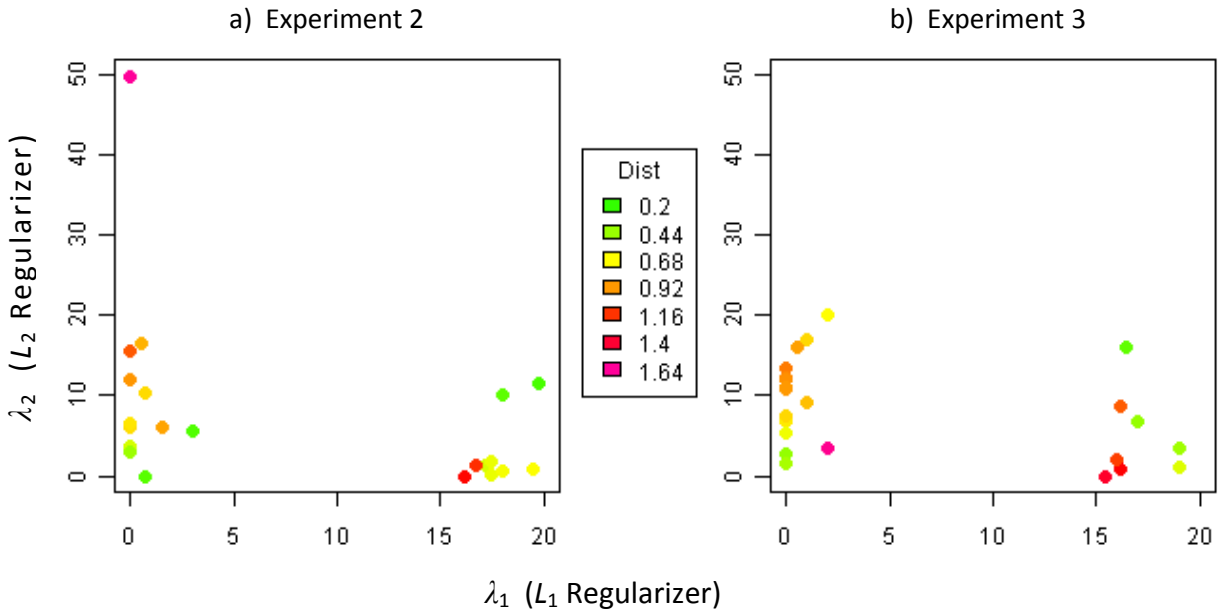
Non-sparse regularizers (e.g. L_2) shrink β weights, especially larger ones—therefore, their points tend towards the origin of the graph.

It can be hard to tell by looking at a cloud of participants' βs whether any deviation from the true β is towards the axes or origin—these two patterns look very similar for small divergences from the true β, and the participant β cloud is highly varied and noisy itself. In order to rigorously estimate what regularizer best describes each participant's deviation from the true β, I used a variation of tuning an elastic net (linear regression using both L_1 and L_2 regularization; see equation 6). I repeatedly performed the elastic net regression analysis using the same training set $x_{1,2,3}$ and y that participants learned with, incrementing λ_1 and λ_2 slightly until I had grid-sampled a large number of possible λ settings. However, instead of determining the best model based on performance on some tuning set, I looked to see what model was closest to the model developed by each participant, thereby finding the best-fitting λ_1 and λ_2 for each individual participant.

"Closest" was defined as the (Euclidean) distance between the model's β vector (generated by the regression analysis), and the β vector I calculated for the given participant using the paired trials probe method discussed in Experiment 1 ("Results: Estimating Human β"). Therefore, I were simply looking for the combination of λs that would produce the same (or closest possible) βs as what I estimated for that participant. The distance would not be zero, however, meaning that participants' βs cannot be explained by L_1 and L_2 regularization alone. However, the distances were still quite small (M = 0.76, SD = 0.35), and it's uncertain to what extent nonzero distance should be attributed to some third cause, or simply noise and measurement error.

Interestingly, the fitted λs clearly divide participants into two clusters (Figure 5): those with λ_1 less than 5 and those with λ_1 greater than 15. Because λ_1 represents the relative

Figure 5: Fitted λs



Two-dimensional representation of participants' λs, found through tuning an elastic net, for Experiments 2 and 3. Each point represents an individual participant, and is color-coded to indicate the Euclidean distance from the model-generated β for those λs to the participant's actual β.

importance of sparsity (as it is the weight of the L_1 regularizer in the expression minimized by the regression analysis), I call the clusters the "non-sparse" and "sparse" strategies, respectively. In fact, not only is λ_1 lower for participants with non-sparse strategies (M = 0.64, SD = 0.90) than those with sparse strategies (M = 17.65, SD =1.22) (t = -40.17, p < .001), λ_2 is *higher* for non-sparse strategists (M = 10.29, SD = 11.79) than sparse ones (M = 2.79, SD = 4.31) (t = 2.25, p = .04), further validating the claim that the first cluster is of participants using (relatively) non-sparse regularization and the second is of those using sparse regularization. Importantly, there are no differences in terms of the Euclidean distances between the βs produced by fitting and the participants' actual βs between the non-sparse (M = 0.77, SD = 0.33) and the sparse strategy clusters (M = 0.75, SD = 0.39) (t = 0.13, p > .1).

Despite representing different regularization strategies, participants in the non-sparse cluster (M = 5.96, SD = 3.73) performed just as well in the testing phase (in terms of SSE) as participants in the sparse cluster (M = 6.82, SD = 1.59) (t = -0.79, p > .1). Finally, removing the outlier in the non-sparse cluster with $\lambda_1 = 0$ and $\lambda_2 = 49.71$ has no effect on any of these findings.

Discussion

The two clusters represent two different potential regularization solutions—one favoring the non-sparse (with all-around small β weights) L_2 regularization, and the other favoring sparse L_1 regularization. One reason for the enormous gap in λ_1 values compared to the (still significant) difference in λ_2 values is that a rather high λ_1 is needed to produce a solution with two non-zero βs, while a λ_1 of practically 0 is all that's needed to produce a solution with all three βs being nonzero. Participants using a non-sparse strategy used all three input features, resulting in three non-zero β weights, while those using a sparse strategy were more likely to disregard one of the features. Because there was no difference in performance between the two clusters, this suggests that the sparse strategies were utilizing the redundant input structure: disregarding the independent feature is always detrimental, while if done correctly, only one of the redundant features is needed to achieve the same performance as someone who uses all three correctly.

Since people appear to strongly take one of two approaches to regularizing linear systems, the next logical question is whether we can manipulate which approach people will take. There could be many ways to try to affect this, from asking participants to consciously prefer one strategy, to subtly changing the task to encourage one or the other. Experiment 3 takes the latter tactic, introducing noise to the system in hopes of promoting non-sparse strategies.

Experiment 3

One possible way to manipulate participants' λs is to make the inputs noisy. Thus far, all the inputs have been noiseless: *y* is always exactly the weighted sum of the values of the three input meters. However, noiseless inputs are not realistic: all real-world measurements are fraught with some degree of measurement error, be they due to some sort of measurement device, or the senses of the observer themselves. If all the features are noisy, even if they are redundant it is beneficial to attend to all of them: since redundant features carry the same underlying information, using both is akin to taking repeated measurements and using them to deduce the true value obscured by the noise. Therefore, noisy systems would encourage more *non-sparse* regularization (e.g. L_2). I hypothesized that by replicating Experiment 2 identically except with noisy inputs, I would see a greater preference for L_2 regularization in Experiment 3 than I did in Experiment 2.

Methods

Participants. The participants were 29 undergraduate students (18 females, 11 males) enrolled in an introductory psychology course at the University of Wisconsin – Madison. Ages ranged from 18 to 24 (M = 19.10, SD = 1.23 years). Participation was voluntary, and participants received extra credit for their course for their time. Participants were treated in accordance with the APA's "Ethical Principals of Psychologists and Code of Conduct" (American Psychological Association, 2002).

Materials. The program, computers, and questionnaire were the same as those used in Experiment 1. Once again, participants were run in batches of up to six (M = 4.07, SD = 1.22 simultaneous participants).

Procedure. The procedure was same as the one used for Experiments 1 and 2. Furthermore, the inputs were identical to those used in Experiment 2, except in the training phase random noise was applied to $x_{1,2,3}$ *after y* was calculated. Thus, since β is still (0.55, 1, 1.6), the *y* values remained unchanged from Experiment 2, but the *x* values were slightly different:

$$y_i = 0.55 * x_{1i} + x_{2i} + 1.6 * x_{3i}$$

$$\hat{x}_{1i} = x_{1i} + \varepsilon_{1i}, \ldots \quad (13)$$

Here, $\hat{x}$ is the "noisy" *x* that is actually displayed on the screen—it is the original *x* plus a noise term (ε). (Notice that the equation for *y* still uses *x*, not $\hat{x}$, is used to calculate *y*.) A separate noise term was randomly generated (uniform in (-0.075, +0.075)) for each *x* value for each example. This relatively small range of noise was selected because it produced the highest degree of difference between L_1 and L_2 regularizations' βs, for reasonably small λ values. (For very large λ values, L_1 and L_2 regularization will always produce drastically different solutions, as the regression will emphasize the regularization term well over training set accuracy.) Because x_1 and x_2 have different, independent noise terms applied to them, the strict redundancy relationship between x_1 and x_2 is broken; however, ε is relatively small, so they are still highly intercorrelated. Noise is *not* applied in the testing phase, making those inputs identical to Experiment 2.

Results

No effect of participant age, sex, study time, number of participants run simultaneously, or which computer participants used on any of the following analyses was found. As before, non-learners were defined as those who did not achieve an SSE < 12 in the testing phase: 4

participants were classified as non-learners and were excluded from the following analyses, leaving $n = 25$.

Performance. As in Experiment 1, accuracy over time in the training phase is reported using average squared error (Figure 3). Unlike in Experiment 1, only one participant reached criterion and finished training early. Performance in the testing phase was measured using the SSE for all 23 test trials ($M = 5.22$, $SD = 2.00$).

Estimated Human β. As in Experiment 2, participants' βs were lower than the true β for all three β weights (Figure 4). β_1 was below the true 0.55 ($M = 0.41$, $SD = 0.28$, $t = -2.48$, $p = .021$), β_2 was below the true 1.00 ($M = 0.68$, $SD = 0.27$, $t = -5.92$, $p < .001$), and β_3 was below the true 1.60 ($M = 0.89$, $SD = 0.20$, $t = -17.52$, $p < .001$). Also as with Experiment 2, only participants' estimated β_3 was correlated with testing phase SSE ($r = -.73$, $p < .01$), and the correlation was again negative—the lower β_3 was, the worse that participant performed.

Fitted λ and Cluster Comparison. λ_1 and λ_2 were fitted using the method described in Experiment 2 ("Results: Determining the Human Regularizers"). The Euclidean distance between a given participant's β and the β produced by their best-fit λs was sufficiently small ($M = 0.84$, $SD = 0.34$). Once again, participants were sharply divided into non-sparse and sparse clusters (Figure 5). Just as in Experiment 2, λ_1 lower for participants in the non-sparse cluster ($M = 0.41$, $SD = 0.72$) than participants in the sparse cluster ($M = 16.74$, $SD = 1.36$) ($t = -39.56$, $p < .001$), λ_2 is higher in the non-sparse cluster ($M = 9.50$, $SD = 5.57$) than in the sparse cluster ($M = 4.33$, $SD = 5.34$) ($t = 2.26$, $p = .03$), and there are no differences in the distances between the fitted models' βs and participants' βs between the non-sparse ($M = 0.82$, $SD = 0.27$) and the sparse clusters ($M = 0.89$, $SD = 0.44$) ($t = -0.43$, $p > .1$).

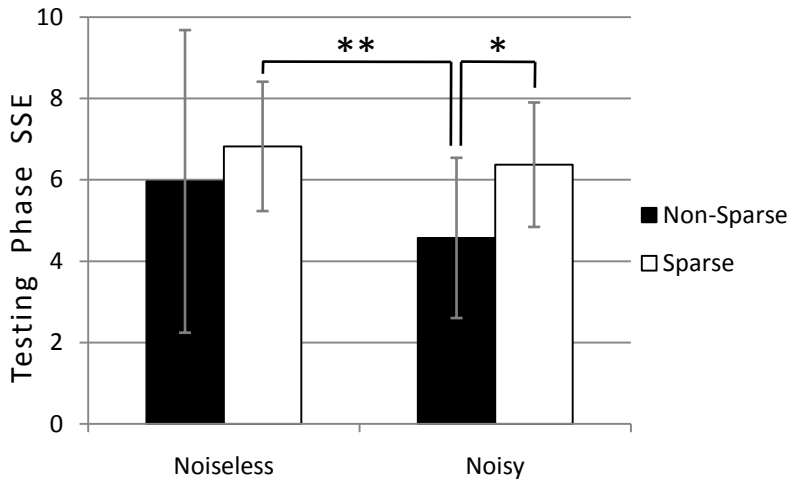
Curiously, *unlike* in Experiment 2, participants in the non-sparse cluster (M = 4.57, SD = 1.97) performed better in testing phase (had lower SSE) than participants in the sparse cluster (M = 6.37, SD = 1.53) (t = -2.36, p = .03). (For further discussion, see "Effect of Noise.")

Effect of Noise. To establish whether the introduction of noisy input features (x values) has the hypothesized effect of promoting non-sparse regularization, I compared the data from Experiments 2 and 3. Because these experiments were conducted about two weeks apart, first they must be checked for any confounding demographic differences. (Note: all statistics reported here are for learners only.) There were no differences between the experiments for participant age, sex, number of participants run simultaneously, or which computers participants chose to use. Participants were run earlier in the day for Experiment 2 (M = 10:19 am, SD = 4.61 hours) than Experiment 3 (M = 11:46 am, SD = 0.59 hours) (t = -1.57, $p < .001$). However, it is unclear whether this would have had any impact on the data, or indicates any averse sampling effects.

Regarding performance, participants in Experiment 2 actually had performed more poorly (had higher SSE) during the testing phase (M = 6.31, SD = 3.04) than participants in Experiment 3 (M = 5.22, SD = 2.00) (t = 1.49, p = .03). Interestingly, using the same t test method used to compare the positive and negative conditions in Experiment 1, no differences were observed between participants' βs in the two experiments (β_1: t = -0.23, p > 0.1; β_2: t = 1.56, p > 0.1; β_3: t = -0.35, p > 0.1). This hints that participants did *not* develop different solutions despite the noisy inputs.

Further comparison between the fitted λs of the two experiments also did *not* support my hypothesis that noisy inputs would promote non-sparse solutions: the ratio of participants that fell in the sparse versus non-sparse cluster was the same in Experiment 2 (M = 40%, SD = 50.0% are *sparse*) and Experiment 3 (M = 36%, SD = 49.0%) (t = 0.29, $p > .1$). There were also no

Figure 6: Testing Phase Performance by Noise and Strategy



Comparison of testing phase performance (measured by SSE) between participants who adopted non-sparse versus sparse regularization strategies (determined by which λ cluster they fell into), and noiseless versus noisy inputs (Experiments 2 and 3 respectively). * denotes a difference with $p < .05$, and ** denotes a difference with $p < .01$.

differences between participants in the non-sparse cluster in Experiments 2 versus 3 in terms of their fitted λ_1, λ_2, or the distances between the fitted models' βs and participants' βs (all $p > .1$). The same is true of participants in the sparse cluster of Experiment 2 versus 3 (all $p > .1$).

A two-way ANOVA analyzing the effects of noisiness (Experiment 2 or 3) and regularization strategy (non-sparse or sparse cluster) on testing phase performance (in terms of SSE) revealed only a partial effect of strategy ($F = 3.24$, $p = .08$) (all other effects' $p > .1$) (Figure 6). Specifically, participants with a non-sparse strategy in a system with noisy inputs (Experiment 3) ($M = 4.57$, $SD = 1.97$) outperformed participants with a sparse strategy in both noisy (Experiment 3) (see "Fitted λ and Cluster Comparison") and noiseless systems (Experiment 2) ($M = 6.82$, $SD = 1.59$) ($t = 3.03$, $p < .01$).

Discussion

My hypothesis that participants would adjust their regularization strategies to favor non-sparse solutions in systems with noisy input was not validated. However, because participants with non-sparse solutions outperformed those with sparse solutions, and even those with sparse solutions in the noiseless version of the experiment (which is impressive, since the noiseless version should be easier for participants), it is clear that those who *did* adopt a non-sparse regularization strategy benefited from it. Why, then, did more participants not favor this strategy in the noisy experiment? Is it that people are inflexible, and some people will always prefer non-sparse solutions and others will always prefer sparse ones? This seems unlikely. However, I cannot determine from these experiments alone whether people are able to adjust their strategies at all, and if so, what would trigger this adjustment.

General Discussion

Summary

Little is understood about humans' ability to learn linear regression tasks; and even less is understood about how the presence of redundant or intercorrelated input features affects their learning. What previous work there is suggests that redundant data actually *impairs* humans' learning, an unexpected result from a mathematical and computational point of view. One possible explanation for this detrimental effect could be that humans regularize their solutions to linear systems, perhaps even more so when their training set appears to involve redundancy. In the carefully designed training examples of psychological studies, regularization might hurt performance with no apparent benefit. Therefore, to better understand how people respond to redundancy, I sought to explore how people regularize their solutions to linear systems.

Experiment 1 acted as a baseline: I established that participants could learn in our paradigm, and that I could estimate their βs using our paired trials probing method. Unfortunately, although the true β of (1,1,1) was easy for participants to learn, it is a "stacking" solution, meaning all the input meters' values are simply summed to find the answer. This seems to be a narrow corner case, and invalidates any attempt to discern what regularizers participants are using. In Experiments 2 and 3, a true β that was far from any stacking solution was used, and I used a tuning method to fit the λs for elastic net regression to each participant individually to determine to what extent they were using L_1 and L_2 regularization. I found that participants fell into two groups: non-sparse and sparse regularizers. To test the adaptability of humans, in Experiment 3 I introduced noise to the inputs: however, this did not appear to alter participants' strategies despite those adopting the non-sparse strategy clearly outperforming those with sparse strategies.

Implications for Psychology and Machine Learning

This study revealed that humans do regularize their solutions to linear systems, and furthermore, they are able to use both sparse and non-sparse regularization. Interestingly, while may use a mixture of the two, they still strongly adopt one strategy or the other, only using the opposite regularizer lightly in comparison to their primary regularizer. Unfortunately, it is uncertain how stable versus task-related people's preference for sparse or non-sparse regularization is—they did *not* appear to adapt their strategy to a noisy situation, despite the fact that those who ended up using a non-sparse strategy performed better.

I also found evidence that people, at least unconsciously, noticed and even made use of the redundancy between two of the input features. For example, in the Experiment 1, the only difference between the positive and negative conditions was the "sign" of the redundancy relationship (whether the redundant features always had the same sign (were both positive or both negative), as in the "positive" condition, or had opposite signs, as in the "negative" condition). However, this simple change to the redundancy relationship caused participants in the negative condition to learn more slowly in the training phase but then outperform the positive condition participants in the testing phase, suggesting that the redundancy structure was affecting participants' learning and solutions. Furthermore, in Experiment 2, participants who adopted a sparse regularization strategy performed equally well in the testing phase compared to those who adopted a non-sparse strategy, suggesting that they learned to attend to one of the redundant features less, *not* the independent feature (as paying less attention to the independent one would invariably reduce performance.) (On a side note, this is perhaps why only participants' estimated β weight for the independent feature was correlated with performance.)

These findings have strong, if still ill-understood, implications for both psychology and machine learning. That humans regularize their solutions to linear systems at all is a previously unverified finding (Schmitt & Dudycha, 1975; Pishkin & Williams, 1984; Hutchinson & Gigerenzer, 2005). With regards to psychology, this suggests that humans actively take measures while learning to improve the generalizability of their rule, even if it involves sacrificing performance during the learning stage itself, something not always considered or discussed when investigating human learning. For example, regularization could be used to explain some earlier conflicting results, such as how humans perform *worse* when feature redundancy is present (Karelaia & Hogarth, 2008)—it might be that they are carefully regularizing the redundant data, only to sacrifice performance during the psychological experiment itself! It may be the case that, in fact, redundant inputs do not confuse or impair humans, but simply cause them to respond differently and more prudently by regularizing.

Similarly, those involved in machine learning should be interested to know that humans actually use some of the methods statisticians and machine learning theorists have developed and used to avoid overfitting. A question raised but not satisfyingly answered by this study is whether humans adapt their regularization strategies to suit the task at hand, something machine learning systems usually do *not* do. Given further research into this question, we might be able to develop rules for when what strategies should be used, and how they can be best combined, helping machine learning researchers to develop computer learners who are as adaptive and dynamic as humans, a continual goal in the field.

Potential Issues and Flaws

The data generated by this study have several seemingly paradoxical findings. For example, while participants clearly divide into two groups for their fitted λs, there were no observed differences between the estimated βs for these groups. This indicates a potential flaw in either the human β estimation method, or the λ fitting method. Another paradoxical finding is that participants' estimated βs are not always correlated with their testing phase performance. Of course, as was at least the case in Experiments 2 and 3 (where only β_3, the independent feature's β, was correlated with performance), this could be evidence that participants noticed and used the redundant relationship between two of the inputs. (See "Implications for Psychology and Machine Learning.") Perhaps on a related note, in Experiment 2, sex was a predictor of test SSE, but not of their estimated β or fitted λs. Clearly, more investigation needs to be done into the paradigm itself, to understand how these measures are interacting, and if they are indeed valid measures.

Suggestions for Future Study

The current work is only the beginning of this line of research. *Many* additional questions have been raised, some of which have already been mentioned. Some are simple, such what effect increasing the number of features (both over all and redundant) would have, or if a different computer interface or "game" is used. One could also explore using a categorization task instead of a continuous-valued judgment task (i.e., ask for discrete (even binary) answers instead of a continuous output measure), or using categorical inputs instead of continuous inputs.

One major line of questioning would be to find out if there is a way to manipulate humans' λs, to test the extent of their situational adaptability. In addition to simply replicating

Experiment 3 with even noisier inputs, one could take a sort of opposite approach and severely limit the time participants have to process the inputs, to encourage sparse strategies. For example, the inputs could be only briefly flashed on the screen. The timing could be chosen such that participants don't have enough time to adequately read all of the input meters. While this would obviously force them to a more sparse solution, the question is whether they can find the optimal sparse solution (the one L_1 regularization would settle on.) If there are strictly redundant features, the hope would be that participants learn to ignore of these redundant inputs. A different approach to testing human adaptability could be to explore how well people can consciously control the sparsity versus non-sparsity of their β solutions. This could be as simple as explaining these concepts to them beforehand, or involve some sort of subtle variation in the description of the task at hand.

Another possible line of questioning is about the so-called "third" regularizer: instead of preferring to have the fewest nonzero β weights (sparse regularization) or the smallest β weights (non-sparse), a regularizer which prefers the "easiest" β weights. This might include "stacking" solutions that allow people to add up the x values without any scaling (e.g. a β of (1,1,1), (1,1,0), (1,0,1), etc.) Other "easy" βs could be those where the xs can be summed and then the whole sum scaled, because all the β weights are equal (e.g. β of (2,2,2)). Furthermore, it is unclear whether stacking solutions are easy in general, or because of the particular design of the user interface for out paradigm. It would be prudent to explore stacking using a different interface. While it would be relatively simple to come up with a regularizer term that penalizes β weights that aren't "easy" enough, one would first need to carefully explore what exactly constitutes "easy." This could be done by grid-sampling the three-dimensional β space, to try and find βs that are easier for humans to learn than others.

Works Cited

Amaldi, E. & Kann, V. (1998). On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theoretical Computer Science, 209,* 237-260.

Brunswik, E. (1955). Representative design and probabilistic theory in a functional psychology. *Psychological Review, 62,* 193-217.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). glmnet: Lasso and elastic-net regularized generalized linear models [Computer software and manual]. Retrieved May 18, 2011, from http://cran.r-project.org/web/packages/glmnet/index.html

Hutchinson, J.M.C. & Gigerenzer, G.(2005). Simple heuristics and rules of thumb: Where psychologists and behavioural biologists might meet. *Behavioral Processes,* 69, 97–124.

Karelaia, N. & Hogarth, R.M. (2008). Determinants of linear judgment: A meta-analysis of Lens Model studies. *Psychological Bulletin, 134,* 404-426.

Pishkin, V. & Williams, W.V. (1984). Redundancy and complexity of information in cognitive performances of schizophrenic and normal individuals. *Journal of Clinical Psychology, 40,* 648-654.

Schmitt, N. & Dudycha, A.L. (1975). A reevaluation of the effect of cue redundancy in multiple-cue probability learning. *Journal of Experimental Psychology: Human Learning and Memory, 104,* 301-315.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of Royal Statistical Society: Series B, 67,* 301-320.

Appendix A – Experimenter Protocol

Hello, everyone. Thank you for coming to our study today. The study will last for an hour and you will receive two participation credits for your time.

This study is investigating how people develop mental rules for relating input and output values. You will be playing a simple game in which you are a palaeontologist-in-training, learning how to use the results of chemical lab tests to determine the age of fossils.

Before we begin, we need to receive your signed consent as a participant in this study. You should all have a consent form at your station. If you have not already, please *carefully* read the entire consent contract before signing. If you have any questions, feel free to ask me.

[Wait until all consent forms have been signed and collected.]

Before you begin the experiment, let me quickly describe how the game works: you will be shown a fossil and the results of the lab tests done on that fossil. The test results give you the relative levels of a chemical marker in different parts of the fossil's body. This chemical is used to determine the age of the fossil. *The catch is that the level of the chemical in some parts of the body is more important than its level in others.*

A screen will appear with the results of the lab test in the form of three red meters, plus a slider in the middle, and a timer in the lower left. Use the up and down arrow keys on the keyboard to set the slider to what you think is the fossil's age. The timer in the lower left will start with 10 seconds on it, and once it hits zero, your answer will be submitted whether you were ready or not. Therefore, be sure to set the slider to the value you want before the time runs out.

After the timer runs out, you will be shown the "Results" screen, where you will again see the three red chemical meters for the original lab results, as well as the guess you made in blue and the official age of the fossil next to it in green. Any difference between your guess and the official age is called your "error", and gets added to a little error bar on the far right. Like a game of Tetris, filling this bar is a bad thing. If you can get through 20 examples *without* filling the error bar, you're done with your palaeontology training! On the other hand, if the error bar ever fills up because you keep having large errors, or differences between your answer and the correct one, then the bar will reset, but so will your count towards the 20 turns.

Try to get your guesses about the ages to be *as accurate* as possible; as you see more and more fossils, you'll get a better feel for the relationship between the chemical readings and the fossil's age. If you do well enough and keep your error bar low, you'll complete your palaeontology training and go on to the final exam. The final exam is just like the rest of the game, except you no longer get to see the correct answer after each guess.

If you didn't just memorize all that, don't worry--the directions will appear on your screen again before the the experiment begins, and you can read through them at your own rate. And you can *always* ask me questions at any time during the experiment.

Before we begin, are there any questions right now? *[Address questions.]* Once you are ready, review the directions, and begin!

Appendix B - Questionnaire

Try to *thoroughly* describe the "rule" you developed for determining fossils' age:

_____

_____

_____

_____

_____

On each of the blanks below, rate **how important** the red meter that was in that position was to determining your answer on a scale of **0 to 100**, where 0 is you did not use that meter at all. *Note that more than one meter can have the same importance rating,* i.e., if you think they were tied!

_____
2

_____
1

_____
3

Now, consider ONLY the red meter on the far left (meter #1 in the diagram above). *In at least one full sentence*, clearly describe the affect that meter had on your answer:

_____

_____

_____

Answer the same question for the top-most meter (meter #2 in the diagram):

_____

_____

_____

Answer the same question for the bottom-most meter (meter #3 in the diagram):

_____

_____

_____

Share you thoughts! Let us know what you thought of this study, or anything else you'd like us to know (like you're very sleepy today, or were confused or bored):

_____

_____

_____

_____

Please provide us with the following demographic information. This data will not be used to identify individual participants, only to describe the sample population we used for the study.

Age: _____

Sex: ☐ Female ☐ Male